

DATABASE Expert

Relationship problems are on the minds of *Shopper* readers, so Kay Ewbank lends a sympathetic ear to a couple who need help with relational database identities and matching the contents of two combo boxes



MYSQL 5.0 TO DOWNLOAD

The first Release Candidate of MySQL 5.0 is available for download. Users have already downloaded previous beta versions two million times and the developers feel the version now available for download is close to the final product. It is available at <http://dev.mysql.com/downloads/mysql/5.0.html>.

New ANSI SQL standard features have been introduced in the latest release, many of the existing features of MySQL have been made ANSI-compliant and there are new MySQL Storage Engines, tools and extensions. New ANSI features include Views, Stored Procedures and functions that use the SQL 2003 syntax, row-level triggers and server-side cursors.

The new engines, tools and extensions include an archive storage engine for storing large amounts of data without indexes, which is useful for future auditing, and an Instance Manager that can be used to start and stop MySQL Server remotely.

It's still early days, though, and the software is not yet suitable for production systems. MySQL's Kaj Arnö said: "We haven't crossed the finish line yet, so we still very much need your involvement to ensure that MySQL 5.0 is the best that it possibly can be".

■ QUERY BUILDER 2005 ARRIVES

EMS Query Builder 2005, a component suite for Borland Delphi and C++ Builder, has been released for download. The suite is designed to give users of applications developed in Delphi or C++ Builder the means to design their own SQL statements for Select, Insert, Update and Delete clauses. It can be used to build new queries visually and to represent existing queries in applications graphically.

The suite includes components for working with standard SQL, Microsoft SQL Server, InterBase/FireBird, MySQL and PostgreSQL. The component supports queries with unions and sub-queries.

A free trial version is available from www.sqlmanager.net and the full version costs £135 including VAT.

Q I have built a relational database in Access where each table has a unique identity and each identity is related to other tables where necessary, with cascade update/delete. The relationships are one-to-many. The problem is Access won't let me enter data into any of the tables because the related tables don't yet have a unique identity in place. The message reads: "You cannot enter data into this table at the moment because table xxx is connected to it and needs a unique entry into connected field."

How do I get information into all the connected fields at once? I have tried making up a query with several fields from the different tables, but either the same message appears or the table does not allow updates.

James Howley

A The problem is you have tables with a one-to-many relationship, so the record on the 'one' side must exist before you can refer to it on the 'many' side. The difficulty you're experiencing is probably down to the order you are entering data in your forms. It's all too easy to try putting information into the related tables the wrong way up.

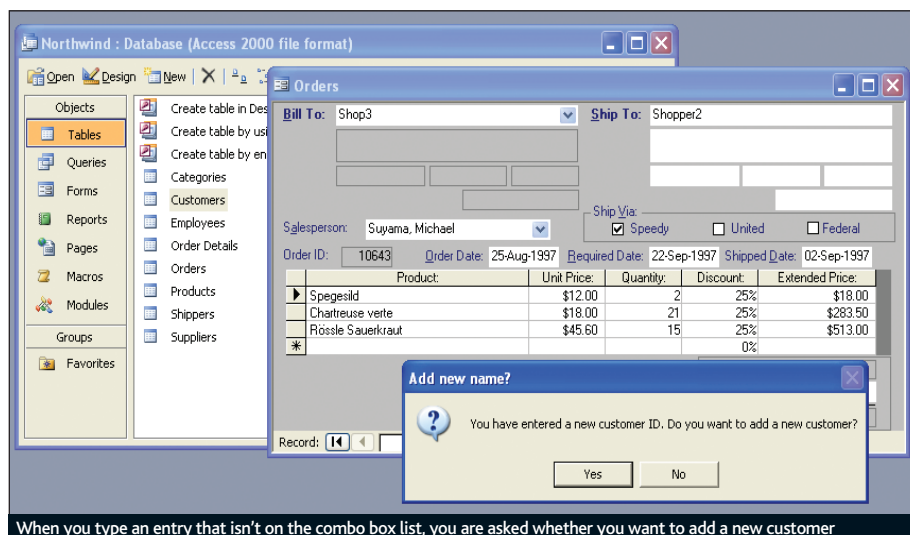
For example, if you were interested in creating an order-processing database, you might start off thinking in terms of orders placed. For a new customer, you'd think in terms of a new order, then the customer to whom the order relates. You need to set up your forms so the customer is created first so you can't accidentally create an order that doesn't belong to a non-existent customer.

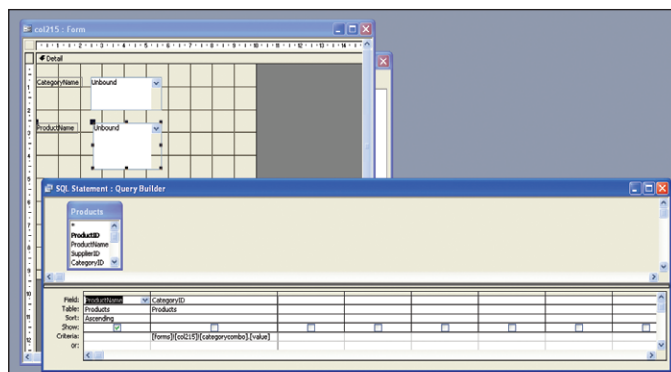
To make it easy to enter the data in the more natural way, by creating the order and then

choosing the customer from a list, it's easiest to use a combo box to display a list of existing customer names you can choose from.

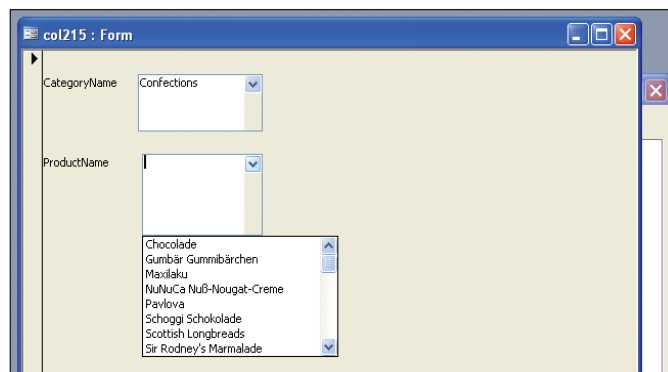
If the order is for a new customer, you should set up your combo box to use the OnNotInList event and add some code along the lines of the sample below. This will work with the Access Northwind sample database, if you want to try it out. Add the code to the combo box in the Orders form, and you will be able to add your own customer details when selecting the customer from the combo box:

```
Private Sub CustomerID_
NotInList(NewData As String, Response
As Integer)
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim strMsg As String, strName As
String
strMsg = "You have entered a new
customer ID. Do you want to add a
new customer?"
If MsgBox(strMsg, vbQuestion +
vbYesNo, "New Customer?") = vbNo
Then
Response = acDataErrContinue
Else
Set db = CurrentDb
Set rs = db.OpenRecordset
("Customers", dbOpenDynaset)
On Error Resume Next
rs.AddNew
rs!CustomerID = NewData
strName = InputBox("what is the
company name")
rs!CompanyName = strName
```





View the combo box properties and click the Build icon to view the query. Set the criterion to check the value from the first combo box



The values that appear in the second combo box change depending on the value you select in the first combo

```
rs.Update
rs.Close

If Err Then
    MsgBox (Err)
    MsgBox "The error above
    occurred. Please try again."
    Response = acDataErrContinue
Else
    Response = acDataErrAdded
End If

End If

Set rs = Nothing
Set db = Nothing
End Sub
```

The code that does the work is pretty simple. Most of the routine is occupied with avoiding errors that will confuse users. The code takes the data that was entered into the combo box by the user. The `NotInList` event is triggered only when the combo box fails to look up the record, so we know the reason the routine is running is because we have new data. First we need to find out whether this is intentional, or whether the user just can't type accurately. A message box asks whether the user wants to add a new customer. If they click No, the routine is closed down and no record is added.

The `Response` argument is what is passed back from the `NotInList` event code, and it has to be one of `acDataErrContinue`, `acDataErrAdded` or `acDataErrDisplay`. The setting indicates how the `NotInList` event was handled. The two of interest here are `acDataErrContinue` and `acDataErrAdded`. `acDataErrContinue` lets you show your own custom message rather than the default message. It is also used as a response from the code to tell Access to carry on as normal. `acDataErrAdded` doesn't display a message to the user, but enables you to add the entry to the combo box list in the `NotInList` event procedure.

After the entry is added, Access updates the list by re-querying the combo box. It rechecks the string against the combo box list and saves the value in the `NewData` argument into the field that is bound to the combo box.

Because the Customers table in Northwind has the `CompanyName` set as required, we also need to provide a value for that column. We can achieve this in our code by asking the user to enter the name using the `InputBox`. Your user will have to enter the other customer details at a later date, including the address, telephone number and so on.

Q I am creating a database in Access 97 where I need to limit the contents of one combo box according to what has been selected in another combo box. This way, I can limit the size of any list that is generated in the second combo box.

I've not managed to find any obvious way of doing this via the usual design wizard. Can you tell me the best way of implementing this, and also if I will need to produce VBA code?
Bob Gilmore

A There may be some way to achieve this without coding, but if you go down the code route the actual programming is quite simple. There are two main methods that you can choose between.

The first is to set up the first combo box using whatever row source you want. Then you set the second combo box's row source to be a query that uses the first combo box as its criterion. We'll use the `Products` and `Categories` tables from the sample Northwind database for my example and create a form that shows category names in one combo box then, depending on the category chosen, displays products from that category in a second box.

First, create a form that is based on the `Products` table, but do not add any controls for displaying data. Add a combo box, call it `CategoryCombo` and set its row source to be the `CategoryName` column from the `Categories` table. You can use the combo box wizard to do this. Choose the option of looking up the values in a table or query, then pick the `Categories` table. Add both the `CategoryName` and the `CategoryID` columns, and sort by the `CategoryName` column. When prompted, accept the default of hiding the `ID` column. Doing this can make combo boxes more confusing to work on, but it looks better.

Next create a second combo box. Using the wizard, choose the option that picks the values from a table or query, and choose the `Products` table as the data source. Pick the `ProductID` column, the `ProductName` column and the `CategoryID` column. Sort by `ProductName` ascending and accept the hiding of the `ProductID` column.

Then view the combo box Properties. Change the name of the combo to `ProdCombo`. Click on the build icon (the one with three dots) on the `RowSource` property, and you'll see a query builder window. In the column for `CategoryID`, enter this criterion:

```
[forms]![col215]![categorycombo].
[value]
```

Change the name of the form from `col215` to whatever you've called your form. You can also remove the tick for displaying the column.

We then come to the code that changes the values in the second combo box, based on the choice you made in the first. In the first combo, click on the `AfterUpdate` event and choose `Code Builder`. The code you need to enter is as follows:

```
Private Sub categorycombo_
    AfterUpdate()
        Me.ProdCombo = Null
        Me.ProdCombo.Requery
    End Sub
```

When you make a change to the value selected in the category combo, you set the value of the `ProdCombo` to `Null`, then re-query it to populate it with the new values that match the category you've chosen. If you've set up the combo boxes to be bound to the data on the form, you'll need to add the same two lines of code about `ProdCombo` to the form's `Current` event.

There is an alternative method that achieves the same result. Instead of setting the `ProdCombo` to `Null` and then re-querying it, you can set its row source in SQL in the `AfterUpdate` event of the first combo box. So your code would look like this:

```
Private Sub categorycombo_
    AfterUpdate()
        Dim sSQL as String
        sSQL="SELECT Products.ProductName "_
        & "FROM Products WHERE" _
        & "(((Products.CategoryID)=forms!
        col215!categorycombo.value))" _
        & "ORDER BY Products.ProductName"
        Me.prodcombo.RowSource=sSQL
    End Sub
```

We've used the `Me` keyword in the VBA code, but not in the SQL string, because it's not a concept SQL recognises. If you try using `Me`, it will be treated as a strange parameter and you'll be asked for the value when you run the query or, in this case, when you move to the second combo. **CS**

CONTACT

KAY EW BANK

Kay is Computer Shopper's database expert and has worked as a consultant and author for 20 years

kay@computershopper.co.uk